Project 2 report

Table of Contents

1. Overview

1.1. Members

Nguyễn Đức Mạnh - 20125012
Nguyễn Hữu Hào - 21125160
Huỳnh Bá Đông Cát - 20125087

1.2. Assignment plan

Tasks	**Assign to**	**Work percentages**
Design pretty UI	Huỳnh Bá Đông Cát	33%
Algorithm Implementation	Nguyễn Hữu Hào	33%
Writing report	Nguyễn Đức mạnh	33%

2. Environment and How to set up

Step 1: Get the project source
clone the project from https://github.com/Boorgar/CS420_Project02/tree/main
Command: git clone git@github.com:Boorgar/CS420_Project02.git

You will need python 3 to run the project and also pip to install dependencies

Step 2: Install dependencies
Get **Pygame** from pip
Command: python3 -m pip install -U pygame –user

Step 3: Make sure to open your console in directory containing these files:
main.py – console version
Graphic.py – graphic version

Step 4: Run the executable
Run the project by one of following commands:
python3 main.py
python3 Graphic.py

3. Demonstrations

```
C:\Users\manhh\OneDrive\Desktop\hcmus\AI\CS420_Project02>python3 main.py
pygame 2.5.2 (SDL 2.28.3, Python 3.11.7)
Hello from the pygame community. https://www.pygame.org/contribute.html
World map printed to Output.txt
path: [(0, 0), (1, 0), (0, 0), (0, 1), (0, 2), (0, 3), (1, 3), (2, 3), (1, 3), (1, 2), (1, 3), (0, 3), (0, 2), (0, 1), (
0, 0), (0, -1)]
action list: ['Move Forward', 'Move Backward', 'Turn Left', 'Move Forward', 'Shoot Successfully', 'Move Forward', 'Move
Forward', 'Turn Right', 'Move Forward', 'Move Forward', 'Move Backward', 'Turn Right', 'Move Forward', 'Collect Gold', '
Move Backward', 'Turn Left', 'Move Backward', 'Turn Left', 'Move Backward', 'Move Backward', 'Move Backward', 'Turn Righ
t', 'Turn Right', 'Move Forward']

C:\Users\manhh\OneDrive\Desktop\hcmus\AI\CS420_Project02>
```

Figure 1: Sample of console display of one solution from a test

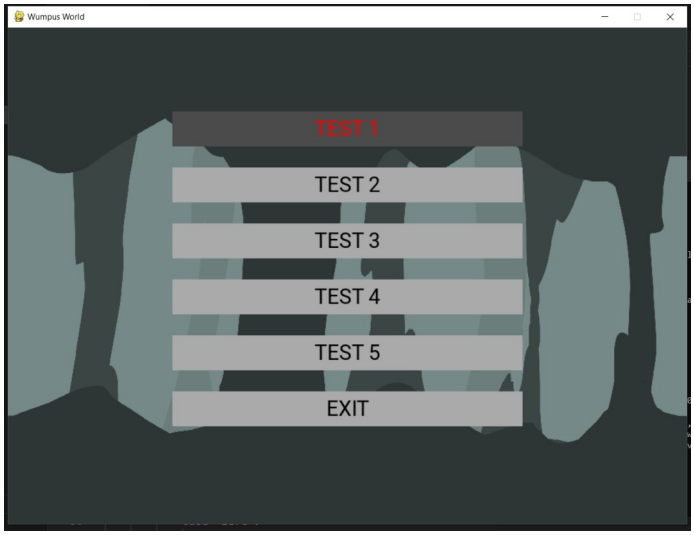

Figure 2: user-interface to select each test case

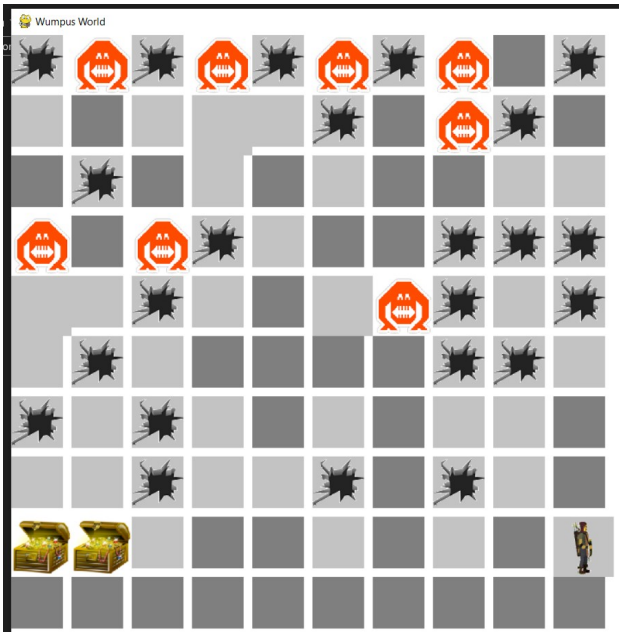

Figure 3: Sample level and animation to demonstrate a possible solution

4. Explaining the Algorithm

4.1. Searching algorithm

The searching method is similar to **DFS**

First it will check for any potential neighbor (except the tile from previous move) and it will go to that tile and save this move to a list, beside of empty tile the agent may find other type of tiles

If the neighbor tile is a **stench**, it will try to shoot the arrow in that direction, there are 2 scenarios:

1. If the **Wumpus** get hit, update the map and the KB
2. No **Wumpus** die, continue to search for others neighbor

If the tile is a **Breeze**, skip for the next neighbor

The neighbor tile is a **gold** step on the tile and collect gold

If there are no valid neighbor, the agent will come back to the tile from previous move

If the agent step on the way out (tile 1, 1) , it will then save this location and the following steps it will take to trace the way out

After the agent choose it target tile (potential neighbor), the it will turn left/right to the direction and do the action relative that way.

→ Repeat the first step until the agent can’t move any more

If it find a way out previously, it will go out now in this step. Otherwise, it will try to kill itself by falling into any **pit**.

4.2. Updating the Knowledge-Base

The knowledge-base is implement in the **KB.py** file and there are some main functions

in_board(x, y) → checking whether the agent currently in valid board coordinate
add_pit(x, y) → update the KB there's a **pit** at x, y
add_wumpus(x, y) → update the KB there's a Wumpus at x, y
add_empty(x, y) → update the KB there's empty tile at x, y
can_t_go_further(x, y) → checking whether tiles is neighbor to x, y is safe to move on
can_find_a_way(x, y) → can find a way by killing any **Wumpus** surrounding x, y
danger(x, y) → is this tile a **Stench** or **Breeze**
kill_wumpus(x, y) → clear the **Wumpus** from that tile and update the KB
get_path_to_door() → return the path to the **exist**
get_previous_action() → get previous action or previous movement
score_history() → total scores

5. Experiments

5.1. Testing

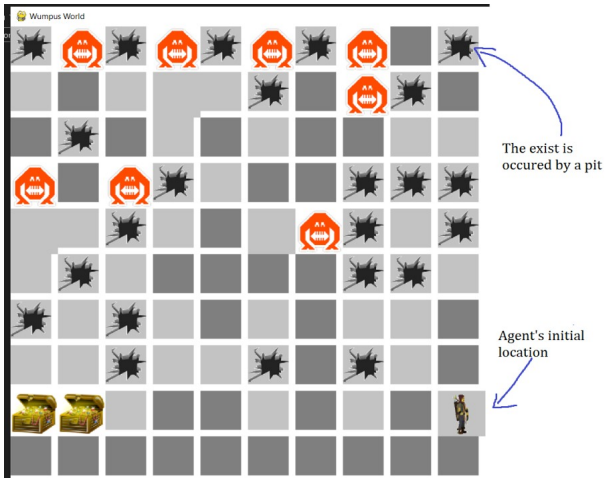

Figure 4: Map from test 3

The Agent successfully collecting all golds.

Agent will scan for all available empty space and gold, but exists is not available; the agent kill itself by falling to nearby pit.

```
Action history ['Turn Right', 'Move Forward', 'Move Forward', 'Move Forward', 'Move Backward', 'Turn Right', 'Move Forward', 'Move Backward'
, 'Turn Left', 'Move Backward', 'Turn Right', 'Move Forward', 'Move Backward', 'Turn Left', 'Move Backward', 'Turn Right', 'Move Forward', '
Move Forward', 'Move Backward', 'Turn Right', 'Move Forward', 'Turn Right', 'Move Forward', 'Move Backward', 'Turn Left', 'Turn Left', 'Move
 Forward', 'Move Forward', 'Move Forward', 'Move Forward', 'Shoot Successfully', 'Move Forward', 'Move Forward', 'Collect Gold', 'Move Forwa
rd', 'Collect Gold', 'Move Forward', 'Turn Left', 'Move Forward', 'Collect Gold', 'Move Forward', 'Move Backward', 'Turn Left', 'Move Forwar
d', 'Collect Gold', 'Move Forward', 'Move Backward', 'Turn Right', 'Move Forward', 'Move Backward', 'Turn Left', 'Move Backward', 'Turn Righ
t', 'Move Backward', 'Turn Right', 'Move Backward', 'Move Backward', 'Move Backward', 'Turn Left', 'Move Forward', 'Move Forward', 'Move Bac
kward', 'Turn Left', 'Move Forward', 'Move Forward', 'Move Backward', 'Turn Right', 'Move Forward', 'Move Backward', 'Turn Left', 'Move Back
ward', 'Turn Right', 'Move Backward', 'Turn Right', 'Move Backward', 'Move Backward', 'Move Backward', 'Turn Left', 'Move Forward', 'Move Fo
rward', 'Collect Gold', 'Move Backward', 'Move Backward', 'Turn Right', 'Move Backward', 'Move Backward', 'Turn Right', 'Turn Right', 'Turn
Left', 'Move Backward', 'Turn Left', 'Move Backward', 'Turn Left', 'Turn Left', 'Turn Right', 'Move Forward', 'Move Backward', 'Move Forward
', 'Move Backward', 'Move Forward', 'Move Backward', 'Move Forward']
```

Figure 5: Possible solution from test 3

5.2. Comments

The Agent successfully scan all empty room and avoid all Wumpus and Pit.

The draw back is if a tile is a Breeze or Stench the agent treat it as block tile and it will not take the risk for possible path.

Overall, the agent is clever enough for most of the cases.

6. Degree of completion

No.	Specification	Scores
1	Finish problem successfully.	50/50
2	Graphical demonstration of each step of the running process.	10/10
3	Generate at least 5 maps with difference structures such as position and number of Pit, Gold and Wumpus	20/20
4	Report your algorithm, experiment with some reflection or comments.	20/20
	Total	100

7. References

- The UI is inspired from kieuconghau/ai-wumpus-world [https://github.com/kieuconghau/ai-wumpus-world/tree/master]